引入隐式反馈的多维度推荐算法 *

刘美博，满君丰，彭　成，刘　鸣

(湖南工业大学 计算机学院, 湖南 株洲 412000)

摘　要：现有的推荐算法大多是应用显示反馈信息来推荐。针对显示反馈信息作出的推荐在准确率和数据稀疏性处理上还存在缺陷的问题，引入了隐式反馈信息，设计和实现了一种引入隐式反馈的多维度推荐算法(iMCF)。该算法涵盖用户、项目和隐式反馈三个维度的信息。对于前两个维度的信息，通过云模型相似度建模；而隐式反馈维度的信息，主要是结合概率矩阵分解模型进行处理。之后再把这三个维度得出的预测评分根据权值进行平衡，得出最终预测评分并作出推荐。实验数据表明，该算法在召回率和准确率上的表现相对于其他算法有了较为明显的提升，且适合大数据环境。

关键词：推荐算法；隐式反馈；多维度；云模型；MapReduce

中图分类号：TP301.6　　doi: 10.3969/j.issn.1001-3695.2018.04.0376

Multidimensional recommendation algorithm with implicit feedback introduced

Liu Meibo, Man Junfeng, Peng Cheng, Liu Ming

(*College of Computer & Communication Hunan University of Technology*, *Zhuzhou Hunan* 412000, *Chian*)

Abstract: The existing recommendation algorithms are mostly applied to display feedback information to recommend. There are still defects in the accuracy of recommendation and data sparse processing for recommendations made by displaying feedback information. Implicit feedback information was introduced, and a multi-dimensional recommendation algorithm (iMCF) that introduces implicit feedback was designed and implemented. The algorithm covered three dimensions of user, project, and implicit feedback. For the first two dimensions of information, it is modeled by the similarity of the cloud model. The information of the implicit feedback dimension was mainly dealt with by combining the probabilistic matrix decomposition model. Afterwards, the prediction scores obtained from these three dimensions were balanced according to the weights, and obtained the final prediction score and made recommendation. The experimental data shows that the algorithm's performance in terms of recall rate and accuracy has been significantly improved compared to other algorithms, and is suitable for big data environments.

Key words: recommendation algorithm; implicit feedback; multidimensional; cloud model; MapReduce

0　引言

网络的迅速普及和通信方式的多样给人们带来了极大的便利，同时也造成了信息量的急剧增加。信息的增多也带来了越来越多的垃圾信息，如何从大量信息中筛选出人们需要的信息就需要用到推荐系统。而现今的推荐系统大多利用的是人们的显式反馈信息[1~4]，如对电影的评分、对商品的评分等。但人们的隐式反馈信息也不可忽略，如用户的浏览记录、购买记录、对相关电影的新闻的关注度等，这些隐式反馈信息获取容易，且信息量较大。因此，结合隐式反馈信息来进行推荐已成为研究热点。

现有的推荐算法中，主流应用的推荐算法就是协同过滤推荐算法，主要可分为基于近邻的协同过滤推荐和基于矩阵分解模型的协同过滤。基于近邻的协同过滤推荐算法主要是通过计算相似度找出最近邻居，再通过最近邻居进行目标用户对目标项目的评分预测，最后作出推荐[5,6]。基于模型的协同过滤现今应用最广泛的就是矩阵分解[7,8]，矩阵分解能够有效地缓解数据稀疏问题，扩展性问题也得到了一定改善，但矩阵分解降维容易造成数据失真，这会导致推荐误差较大、准确率下降。

针对上述问题，通过学习前人的研究成果，本文提出一种

收稿日期：2018-04-22；**修回日期：**2018-06-19　　**基金项目：**国家自然科学基金资助项目(61871432)；湖南省自然科学基金资助项目(2018JJ4063, 2017JJ3065, 2016JJ5035,2016JJ5036)；湖南省教育厅重点项目(16A059, 176A052)；湖南省教育厅优秀青年项目(16B071)；湖南省研究生科研创新项目（CX2017B687）

作者简介：刘美博（1991-），男，湖南株洲人，硕士，主要研究方向为大数据处理（495158123@qq.com）；满君丰（1976-），男（满族），教授，博士，主要研究方向为网络化软件、大数据分析；彭成（1983-），男，讲师，博士，主要研究方向为工业大数据分析；刘鸣（1993-），男，硕士，主要研究方向为大数据处理.

引入隐式反馈的多维度推荐算法。该算法是基于 Hadoop 分布式平台来实现的，并且该算法选取了用户、项目和隐式反馈三个维度的数据展开研究。主要贡献如下：

a)充分利用了用户、项目、隐式反馈三个维度的数据，解决了数据稀疏和单一维度数据量稀少造成的推荐质量不高的问题。

b)有效结合了相似度模型和矩阵分解模型，使得显示反馈数据和隐式反馈数据都得到合理的应用。

c)给出了三个维度的 MapReduce 实现方法，提高了该算法的运算效率和可扩展性。

1　相关研究

1.1　概率矩阵分解模型

概率矩阵分解（PMF）是指在矩阵分解的基础上添加了概率分布[7]。假设有 M 部电影，N 个用户和评分值为 1~K 的评分矩阵。令 R_{ij} 表示用户 i 对电影 j 的评分，$U\in R_{D\times N}$ 和 $V\in R_{D\times M}$ 是用户和电影的隐式特征矩阵，列向量 U_i 和 V_j 分别表示特定用户和特定电影的隐式特征向量。本文采用高斯观测噪声来设计概率线性模型，如图 1 所示。

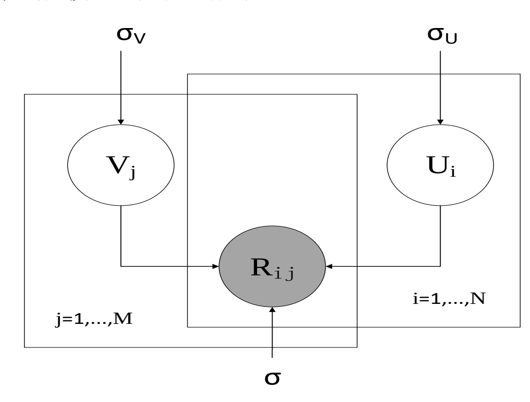


图 1　概率线性模型

将观察到的评级定义为条件分布，如式（1）所示。

$$p(R|U,V,\sigma^2)=\prod_{i=1}^{N}\prod_{j=1}^{M}[N(R_{ij}|U_i^TV_j,\sigma^2)]^{I_{ij}} \tag{1}$$

其中：$N(x|\mu,\sigma^2)$ 是指具有均值 μ 和方差 σ^2 的高斯分布的概率密度函数；I_{ij} 是一个指示函数，当用户 i 对电影 j 进行了评分时为 1，否则为 0。并且还为每一个隐式特征向量添加了一个均值为 0 的球面高斯先验[1,11]，如式（2）所示。

$$p(U|\sigma_U^2)=\prod_{i=1}^{N}N(U_i|0,\sigma_U^2I)$$

$$p(V|\sigma_V^2)=\prod_{i=1}^{M}N(V_i|0,\sigma_V^2I) \tag{2}$$

通过取对数得出用户特征向量和电影特征向量的后验分布，如式（3）所示。

$$\begin{aligned}\ln p(U,V|R,\sigma^2,\sigma_V^2,\sigma_U^2)=-\frac{1}{2\sigma^2}\sum_{i=1}^{N}\sum_{j=1}^{M}I_{ij}(R_{ij}-U_i^TV_j)^2\\-\frac{1}{2\sigma^2}\sum_{i=1}^{N}U_i^TU_i-\frac{1}{2\sigma_V^2}\sum_{j=1}^{M}V_j^TV_j-\frac{1}{2}((\sum_{i=1}^{N}\sum_{j=1}^{M}I_{ij})\ln\sigma^2+\\ND\ln\sigma_U^2+MD\ln\sigma_V^2)+C\end{aligned} \tag{3}$$

其中：C 代表常数。在固定超参数的（即观测噪声方差和先验方差）的情况下，最大化式（3）所示的对数后验概率相当于用二次正则化项最小化公式（4）所示的目标函数。

$$\begin{aligned}E=\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M}I_{ij}(R_{ij}-U_i^TV_j)^2+\frac{\lambda_U}{2}\sum_{i=1}^{N}\|U_i\|_{Fro}^2\\+\frac{\lambda_V}{2}\sum_{j=1}^{N}\|V_j\|_{Fro}^2\end{aligned} \tag{4}$$

$\lambda_U=\sigma^2/\sigma_U^2,\lambda_V=\sigma^2/\sigma_V^2$；$\|\cdot\|_{Fro}^2$ 为 Frobenius 范数,再优化学习式(4)中的 U 和 V 特征矩阵。

1.2　相似度计算

协同过滤推荐算法其基本思想是“物以类聚，人以群分”依据相似性而产生推荐[8,9]。

要想运用协同过滤推荐算法，其数据必须是一个矩阵形式，如表 1 所示。其中 U 是用户，I 是项目，R_{mn} 表示用户集中的用户 m 对项目集中的项目 n 的评分。

表 1　用户—项目评分

	I_1	I_2	…	I_a	…	I_n
U_1	$R_{1,1}$	$R_{1,2}$	…	$R_{1,a}$	…	$R_{1,n}$
U_2	$R_{2,1}$	$R_{2,2}$	…	$R_{2,a}$	…	$R_{2,n}$
…	…	…	…	…	…	…
Ua	$R_{a,1}$	$R_{a,2}$	…	$R_{a,a}$	…	$R_{a,n}$
…	…	…	…	…	…	…
U_m	$R_{m,1}$	$R_{m,2}$	…	$R_{m,a}$	…	$R_{m,n}$

该算法的核心就是寻找最近邻居。要想找到邻居必需计算相似性，本文的度量相似性方法采用云模型相似性度量方法，具体如下。

云模型相似性：先根据评分矩阵统计得出用户或项目的评分频度向量 $U_i=(u_1,u_2,u_3,u_4,u_5)$（1≤i≤m），再通过逆向云算法计算特征向量如下：

$$E\hat{x}=\bar{X}=\frac{1}{N}\sum_{i=1}^{N}x_i\quad \hat{H}e=\sqrt{\frac{\pi}{2}}\times\frac{1}{N}\sum_{i=1}^{N}|x_i-E\hat{x}|\quad \hat{E}n=\sqrt{S^2-\frac{1}{3}H\hat{e}^2}$$

特征向量 V_i=（Ex_i,En_i,Ee_i），V_j=（Ex_j,En_j,Ee_j）。

$$sim(i,j)=\cos(V_i,V_j)=\frac{V_i\cdot V_j}{\|V_i\|\cdot\|V_j\|} \tag{5}$$

通过上述方法计算出相似用户后，在根据式（6）计算来产生推荐。

$$R_{i,d}=\bar{R}_i+\frac{\sum_{j\in NESI}sim(i,j)*(R_{j,d}-\bar{R}_j)}{\sum_{j\in NESI}|sim(i,j)|} \tag{6}$$

其中：NESI 表示邻居集；$\bar{R}_i$ 表示项目 i 的平均评分；$\bar{R}_j$ 表示邻居项目的平均评分。

2　引入隐式反馈的多维度推荐算法

本文主要选取了项目、用户和隐式反馈三个维度数据来综合得出最终预测结果，下面给出具体推荐示意图。如图 2 所示，引入隐式反馈的多维度推荐算法主要由建立用户—项目评分矩

阵、基于用户云模型协同过滤建模、基于项目云模型协同过滤建模、概率矩阵分解建模以及综合推荐等模块组成。建立用户—项目评分矩阵主要是通过对稀疏矩阵进行填充来初步解决数据稀疏问题。之后再通过三个维度的建模来求得个这三个维度的预测评分，并通过权值分配得出三个维度的综合评分，再依据评分来进行推荐。

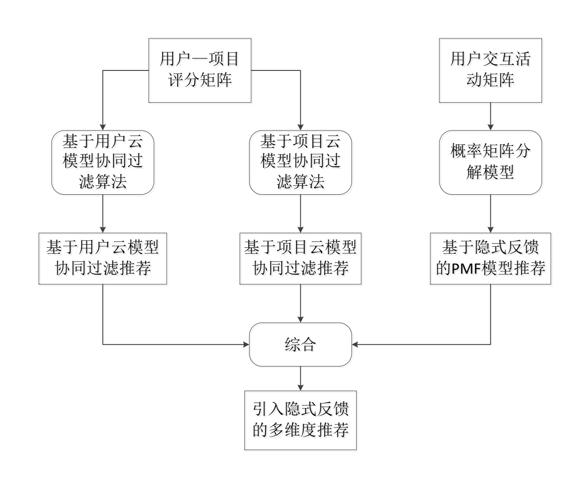


图 2　推荐示意图

2.1　未评分项的预测填充

数据稀疏问题是传统推荐算法普遍存在的问题，会导致相似度计算困难、推荐准确度偏低。本文通过采用云模型相似度来对评分矩阵进行评分预测填充，再利用填充好的评分矩阵进行目标用户对目标项目的评分计算。其有效解决了数据稀疏问题。具体如算法 1 所示。

算法 1　未评分项的预测填充

输入：评分矩阵 $R_{m\times n}$。

输出：预测用户 u 对未评分项目 i_a 的评分。

a) 根据评分矩阵 $R_{m\times n}$，统计出项目的评分频度向量 $\mathbf{I_u}$=(i_1,i_2,i_3,i_4,i_5)（1≤i≤m），再通过逆向云算法计算特征向量 $\mathbf{V_i}$=（Ex_i,En_i,Ee_i）（1≤i≤m）；

b) 根据式（5）计算 i_a 与其他项目的云相似性：

$$\mathrm{sim}(i_a,j)=\cos(V_{i_a},V_j)=\frac{V_{i_a}\cdot V_j}{\|V_{i_a}\|\cdot\|V_j\|}$$

c) 将第二步的相似度值进行大到小排序，得出前若干项目作为邻居集 NESI={$i_1,i_2,\ldots,i_x$}；

d) 得出邻居集 NESI 后，再根据式（6）计算用户 u 对未评分项目 i_a 的评分：

$$R_{u,i_a}=\overline{R}_u+\frac{\sum_{j\in NESI}sim(i_a,j)*(R_{j,d}-\overline{R}_j)}{\sum_{j\in NESI}|sim(i_a,j)|}$$

2.2　三个维度的预测评分计算

基于用户云模型相似度计算第一维度的预测评分：在上面矩阵填充的基础上，根据式(5)计算用户间的相似度 Sim(U_i, U_x)，再使用式（6）求得预测评分，将目标用户 U_i 对目标项目 I_j 的预测评分记为 $UR_{i,j}$，其中目标用户的邻居集为 S（U_i）。

$$UR_{i,j}=\overline{R}_i+\frac{\sum_{Ux\in S(Ui)}Sim(Ui,Ux)*(R_{x,j}-\overline{R}_x)}{\sum_{Ux\in S(Ui)}|sim(Ui,Ux)|}\tag{7}$$

基于项目云模型相似度计算第二维度的预测评分：在上面矩阵填充的基础上，根据式(5)计算项目间的相似度 Sim(I_i, I_y)，再使用式（6）求得预测评分，将目标用户 U_i 对目标项目 I_j 的预测评分记为 $IR_{i,j}$，其中目标项目的邻居集为 S（I_i）。

$$IR_{i,j}=\overline{R}_i+\frac{\sum_{Iy\in S(Ii)}Sim(Ii,Iy)*(R_{i,y}-\overline{R}_y)}{\sum_{Iy\in S(Ii)}|sim(Ii,Iy)|}\tag{8}$$

为了对隐式反馈数据实验的开展，本文选取了从知名影评网豆瓣爬取的数据进行实验，选取了符合条件的 4 万多个用户的数据集，每个人收藏的电影超过 25 部，交互数据超过 30 次。具体实例如表 2、3 所示。

表 2　用户收藏信息示例

ID	收藏的电影
173952145	降临，速度与激情，机械公敌，全民超人，……
209456781	独立日，当幸福来敲门，变脸，空中监狱，……
198746028	荒岛余生，变相怪杰，训练日，木乃伊，……

表 3　用户交互数据示例

电影	简介	点击 IMDB 链接	评论	分享
速度与激情	2	5	3	2
空中监狱	0	0	5	3
荒岛余生	1	3	7	4

针对该数据，借鉴了文献[10]和文献[11]中的相关做法，在此基础上进行了改进。具体方法如下：

①计算用户对电影的隐式兴趣程度向量。设定用户为 u，用户对电影的兴趣信息列表 S（u），用户的交互活动矩阵 A（u），用户收藏的感兴趣电影列表 L（u），S（u）={$s_1,s_1,\ldots,s_n$}，A（u）={$a_{u1},a_{u2},\ldots,a_{un}$}。

$$s_1=\begin{cases}1 & s_i\in L(u)\\ 0 & s_i\notin L(u)\end{cases},1\le i\le n\tag{9}$$

其中：a_{us}（1<=s<=n）是 k 维向量，其中每一个数据为交互类型出现的次数。具体交互类型如下：

a) 用户是否查看电影具体剧情简介；

b) 用户是否点击 IMDB 链接；

c) 用户是否评论和回复他人评论；

d) 用户是否分享该电影。

针对这四种交互类型，本文设定权重系数 α=(1,4,6,10)，再通过式（10）合并交互数据。

$$p_{u,s,i}\begin{cases}1 & a_{u,s,i}\ge threshold\\ a_{u,s,i}/threshold & a_{u,s,i}<threshold\end{cases}$$
$$(1\le i\le k,1\le s\le n),i_{u,s}=\frac{1}{\alpha'}\sum_{j=1}^{k}\alpha_j p_{u,s,j}\tag{10}$$

其中：threshold 为交互次数的三四分位数。然后可得到用户对电影的隐式兴趣程度向量 I（u）={$i_1,i_2,\ldots i_n$}。

②计算用户的兴趣程度评分及评分向量。通过式（11）得出兴趣程度评分。

$$r_{u,s}=(w+(1-w)i_{u,s})s_{u,s} \tag{11}$$

其中：$r_{u,s}$ 为用户对电影的最终兴趣评分；w 为权重参数，取值为 0,85，得出评分向量 R（u）={$r_1,r_2,\ldots r_n$}。

③计算评分矩阵 R 中某一预测评分。通过式（12）得出预测评分。

$$FR_{ui}=\sum_{k=1}^{K}U_{uk}V_{ki}=U_uV_i^T,\quad u\in(1,\ldots,N),i\in(1,\ldots,M),k\in(1,\ldots,K) \tag{12}$$

其中：U 为用户特征矩阵；V 为电影特征矩阵；K 为特征数。而对于未知元素本文使用 AMAN 方法[12,13]设为 0，然后再经过 PMF 模型进行训练得出最终推荐结果。具体示意图如图 3 所示。

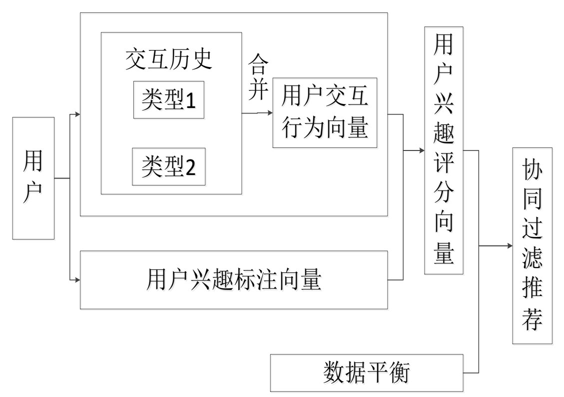


图 3　基于隐式反馈的电影推荐示意图

2.3　最终预测评分

接收以上得出的三个预测评分，动态确定三个评分的权值分配，借鉴文献[17]的方法引入近邻群和信任子群，并按式（13）和（14）得出相关对象集合。

$$S(U_a)=\{U_x\mid Sim'(U_a,U_x)>\mu,a\neq x\} \tag{13}$$

$$S(I_j)=\{I_x\mid Sim'(I_j,I_y)>\nu,j\neq y\} \tag{14}$$

近邻群大小|S(Ua)|=m；|S(Ij)|=n。

$$S'(U_a)=\{U_x\mid Sim'(U_a,U_x)>\mu\ \&\ |I_{U_a}\cap I_{U_x}|>\varepsilon,a\neq x\} \tag{15}$$

$$S'(I_j)=\{I_y\mid Sim'(I_j,I_y)>\nu\ \&\ |U_{I_j}\cap U_{I_y}|>\delta,j\neq y\} \tag{16}$$

信任子群大小|S'(Ua)|=m'；|S'(Ij)|=n'；其中 μ，ν，ε，δ 为阈值。

$$S''(U_a)=\{U_x\mid Sim'(U_a,U_x)>\mu\ \&\ R_{x,j}\neq 0,a\neq x\} \tag{17}$$

$$S''(I_j)=\{I_x\mid Sim'(I_j,I_y)>\nu\ \&\ R_{a,y}\neq 0,j\neq y\} \tag{18}$$

相关对象集合大小|S''(Ua)|=m'',|S''(Ij)|=n''。其中 μ、ν 为阈值。

设 a、b 分别为用户维度和项目维度的权值，则 1-a-b 为隐式反馈维度的权重。

$$a=\frac{\phi\times M}{\phi\times M+\vartheta\times N+q};b=\frac{\vartheta\times N}{\phi\times M+\vartheta\times N+q};\quad 1-a-b=\frac{q}{\phi\times M+\vartheta\times N+q} \tag{19}$$

ϕ、ϑ 是调和参数,q 为 Ua 打过分的项目集大小。M、N 取值如下：

$1.(m''+n'')>0,M=m'',N=n'';$

$2.(m'+n')>0,M=m',N=n';$

$3.(m+n)>0,M=m,N=n;$

$4.M=N=0;$

然后根据式（20）计算目标用户对未评分项目的综合预测评分，算出综合预测评分后，再把评分值进行排序，根据 TOP-N 算法[18]作出推荐。

$$PR_{i,j}=a\times UR_{i,j}+b\times IR_{i,j}+(1-a-b)FR_{i,j} \tag{20}$$

2.4　引入隐式反馈的多维度推荐算法实现

该算法主要可分为以下几个步骤进行具体实现：根据填充后的评分矩阵计算基于用户和基于项目维度的预测评分；根据隐式反馈计算基于隐式反馈维度的预测评分；动态确定权值分配，计算最终的综合预测评分；根据项目最终预测评分高低为目标用户作出推荐。具体如算法 2 所示。

算法 2　引入隐式反馈的多维度推荐算法

输入：已填充的评分矩阵 $R_{m\times n}$，阈值 μ、ν、ε、δ，调和参数 ϕ、ϑ，隐式数据的项目特征矩阵 Q，用户特征矩阵 P，正则因子 λ 值列表(λ_0，$\lambda_1\ldots\lambda_a$)，特征数 L。

输出：为目标用户 U_a 推荐的前 k 个项目编号。

1.　根据式（13）（14）求得用户和项目的近邻群 S（U_a）={$U_1,\ldots,U_s$}和 S（I_j）={$I_1,\ldots,I_t$}；
2.　求出目标用户 U_a 的已评分项目集合 I_{Ua}={$I_1,\ldots,I_q$}；
3.　for j from 1 to n do
4.　If U_a did not rate I_j then
5.　for i from 1 to s do
6.　if U_i rated I_j then
7.　Under1 += USim(U_a,U_i);
8.　On1 += USim(U_a,U_i) * ($R_{i,j}$ - UAvg(U_i));
9.　end for
10.　If　Under1 == 0 then $UR_{a,j}$ = UAvg(U_i);
11.　Else $UR_{a,j}$ = UAvg(U_i) + On1/Under1；
12.　for i from 1 to t do
13.　if U_i rated I_j then
14.　Under2 += ISim(I_j,I_i);
15.　On2 += ISim(I_j,I_i) * ($R_{a,i}$-IAvg(I_i));
16.　end for
17.　If　Under2 == 0 then $IR_{a,j}$ = IAvg(I_j);
18.　Else $IR_{a,j}$ = IAvg(I_j)+ On2/Under2;
19.　for λ in (λ_0,$\lambda_1\ldots\lambda_a$) do
20.　for x in (0,n) do
21.　[$Qu_{x1},\ldots,Qu_{xL}$]←Search(i_x,λ,Q)
22.　$R_{L\times L}$←$(\lambda\times n\times I+Qu^TQu)^{-1}$
23.　for x in (0,L) do

Pux← $\sum_{z=0}^{n}(\sum_{y=0}^{L-1}R_{xy}Qu_{yz})\times r_z$

24.　end for

25.　$[Pi_{x1},\ldots,Pi_{xL}]\leftarrow Search(u_x,\lambda,P)$

26.　$R_{L\times L}\leftarrow(\lambda\times n\times I+Pi^TPi)^{-1}$

27.　for x in (0,L) do

Qix← $\sum_{z=0}^{n}(\sum_{y=0}^{L-1}R_{xy}Pi_{yz})\times r_z$

28.　end for

29.　end for

30.　end for

31.　$FR_{a,j}=P_uQ_i^T$;

32.　calcWeight(a,b,1-a-b)；

33.　$PR_{a,j}=a*UR_{a,j}+b*IR_{a,j}+(1-a-b)*FR_{a,j}$ ；

34.　$Store(P,PR_{a,j})$；

35.　end for

36.　store(P)；

37.　Return Top(k)；

该算法第 5~11 行通过用户相似度计算用户维度的预测评分，第 12~18 行通过项目相似度计算项目维度的预测评分，第 19~33 行通过概率矩阵分解计算隐式反馈维度预测评分，之后通过动态权值分配求得三个维度的权值，再通过计算得出最终综合预测评分，并把该评分降序排列，选出前 k 个推荐给目标用户 U_a。

2.5　推荐过程的 MapReduce 实现

为了方便并行化处理，必须对相关步骤进行 mapreduce 实现。具体如下：

1)对前两个馈维度的处理

①评分矩阵的 MapReduce

Map 阶段：接收输入的<key,value>键值对，其中基于项目这一维度 key 为项目的 ID，value 为用户的 ID 与评分值；基于用户这一维度 key 为用户的 ID，value 为项目的 ID 与评分值。然后进行切分工作，将原始数据切分成若干记录，检查数据是否合理。Shuffle 排序聚集和分发都以 key 值为依据，然后再将合理的数据发送至管道，进入 reduce 阶段。

Reduce 阶段：接收 Map 阶段的数据，根据 key 值进行合并，得到评分矩阵。

②相似度计算的 MapReduce

Map 阶段：接收评分矩阵后，对每个评分数据进行提取，基于项目这一维度以项目对（i_x,i_y）作为 key 值，项目对应的评分对（S_x,S_y）作为 value 值输出。基于用户这一维度以用户对（u_x,u_y）作为 key 值，用户对应的评分对（S_x,S_y）作为 value 值输出

Reduce 阶段：接收 Map 阶段的数据，根据式（5）计算项目间的相似度和计用户间的相似度；将结果保存输出。

③预测评分的 MapReduce

Map 阶段：根据相似度的值，首先基于项目这一维度得出每个项目相似度最高的 N 个项目定义为邻居，以项目为 key 值、项目邻居为 value 值输出。然后基于用户这一维度得出每个用户相似度最高的 N 个项目定义为邻居，以用户为 key 值、用户邻居为 value 值输出。

Reduce 阶段：接收 Map 阶段的数据，根据式（7）和（8）计算出两个维度目标用户对未评分项目的预测评分。

2)对隐式反馈维度的处理

①用户特征矩阵更新的 MapReduce 处理

Map 方法：

输入：（行号 line_key,（电影号 i,用户号 u,对应评分 r））。

输出：（用户号 u,（电影号 i,对应评分 r））。

1.根据行号将（行号 line_key,（电影号 i,用户号 u,对应评分 r））映射为用户号 u

2.if 行号 line_key 小于总行数 M：

3.　输出（用户号 u,（电影号 i,对应评分 r））

Reduce 方法：

输入：（用户号 u,（电影号 i,对应评分 r）对列表）,λ 列表,特征数 k,项目特征矩阵 Q

输出：（（用户号 u,正则因子 λ）,用户特征矩阵 P）

1.for λ in λ 列表 && x in (0，n) do

2.$Search(i_x,\lambda,Q)$得出对应电影特征向量$[Qu_{x1},\cdots,Qu_{xk}]$

3.　计算$(\lambda\times n\times I+Qu^TQu)^{-1}$得出评分值 $R_{k\times k}$

4.　for x in (0,k) do

5.　计算 $\sum_{x=0}^{n}(\sum_{y=0}^{k-1}R_{xy}Qu_{yz})\times r_x$ 得出 P_{ux}

6.输出更新后的用户特征矩阵 U

②项目特征矩阵更新的 MapReduce 处理

Map 方法：

输入：(列号 column_key,(电影号 i,用户号 u,对应评分 r))。

输出：(电影号 i,(用户号 u，对应评分 r))。

1.根据列号将(列号 column_key,(电影号 i,用户号 u,对应评分 r))映射为电影号 u

2.if 行号 line_key 小于总列数 N：

3.输出（电影号 i,（用户号 u,对应评分 r））

Reduce 方法：

输入：（电影号 i,（用户号 u,对应评分 r）对列表）,λ 列表,特征数 k,①得出的用户特征矩阵 P。

输出：（（电影号 i,正则因子 λ）,项目特征矩阵 Q）。

1.for λ in λ 列表 && x in (0，n) do

2.　$Search(u_x,\lambda,U)$得出对应用户特征向量$[Pi_{x1},\cdots,Pi_{xk}]$

3.　计算$(\lambda\times n\times I+Pi^TPi)^{-1}$得出评分值 $R_{k\times k}$

4.　for x in (0,k) do

5.　计算 $\sum_{x=0}^{n}(\sum_{y=0}^{k-1}R_{xy}Pi_{yz})\times r_x$ 得出 Q_{ix}

6.输出更新后的项目特征矩阵 Q

通过①②，分别计算出了用户特征矩阵 P 和项目特征矩阵 Q。再通过式（12）可计算某一预测评分，并通过式（21）计算均方根误差 RMSE[19]。

$$RMSE=\sqrt{\frac{1}{|R|}\sum_{(u,i)\in R}(FR_{ui}-r_{ui})^2} \tag{21}$$

其中：R 为评分集；FR_{ui} 为预测评分；r_{ui} 为评分集中的评分，且 RMSE 与推荐准确度成反向关系。

③均方根误差（RMSE）计算的 MapReduce 处理

Map 方法：

输入：（行号 line_key,（电影号 i,用户号 u,对应评分 r）），λ 列表,特征数 k,①得出的用户特征矩阵 P,②得出的电影特征矩阵 Q。

输出：（λ,ES）。

1.for λ in λ 列表 do

2.　Search(u ,λ,P)得出每一用户特征向量$[u_1,\cdots,u_k]$

3.　Search(i ,λ,Q)得出每一电影特征向量$[i_1,\cdots,i_k]$

4.　计算 $\sum_{x=1}^{k}u_x i_x$ 并设其为 r_a

5.　计算$(r_a-r)^2$ 并设其为 ES

6.　输出（λ,ES）

Reduce 方法：

输入：（λ,ES 列表）。

输出：RMSE。

计算 $\sqrt{(\sum_{x=1}^{n}ES_x)/n}$ 得出 RMSE

1.输出 RMSE

通过①②③的迭代执行，直到 RMSE 不再减小，取最小的 λ 值和特征矩阵，计算出预测评分。通过上述所有算法，再依据 3.4 所示作出推荐。

3　实验及分析

3.1　实验环境与数据度量标准

本文用七台普通的 PC 机搭建 Hadoop 组成集群，命名为 master、slave1~slave6。以豆瓣网站爬取的数据为例，如表 4 所示。

表 4　数据集描述

数据集	用户数	项目数	选择行为
训练集	33 213	84 633	5 095 277
测试集	8 156	23 982	151 972

对数据划分为训练级与测试集，比例大概为 4:1。采用 Rank 值反映召回率（查全率）和 Top-k 命中率反映准确率（precision），命中率、召回率与推荐质量都成正向关系，如式（22）所示。Rank 值表示实际选择的电影在推荐列表中所占百分比位置，实际选择电影在首位，Rank 值为 0%，在最后，Rank 值为 100%。命中率指推荐给用户的 k 部电影，用户实际选择的电影所占的比例。推荐情况示例如表 5 所示。

表 5　推荐情况示例

	推荐	未推荐
喜欢	A	B
不喜欢	C	D

$$召回率=A/(A+C),准确率=A/(A+B) \tag{22}$$

并采用加速比表示处理海量数据时集群节点数对性能方面的影响。加速比定义：$K=T_1/T_n$。其中：T1 表示单节点运行耗费的时长；Tn 表示 n 个节点运行耗费的时长。

3.2　加速性能测试实验

该实验目的主要是针对不同的 TaskTracker 节点，测试该算法在 Hadoop 平台下时效性方面是否显著提升。进行该实验前必须先确定正则因子 λ 值与 RMSE 值，具体取值如表 6 所示。

表 6　不同 λ 值对应的 RMSE 值

λ	0.01	0.03	0.05	0.06	0.07	0.08
RMSE	0.979	0.942	0.923	0.920	0.921	0.923

从表中可以看到，当 λ 值为 0.06 时得到的 RMSE 值最小，所以对 λ 的取值为 0.06。得到 λ 后，再进行加速比实验，分别选取全部数据集和一部分小数据集进行实验，具体结果如图 4 所示。

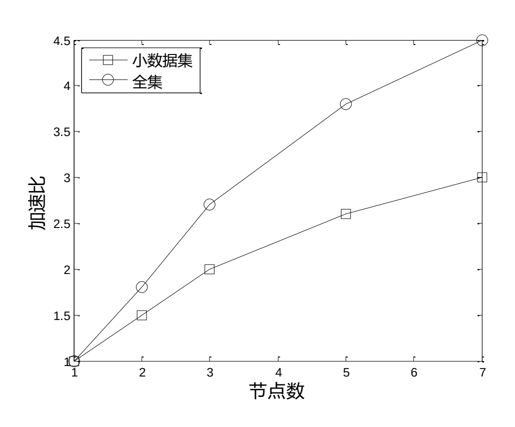


图 4　加速比实验图

从图 4 可看出，当集群节点数从 1 加到 3 时，加速比几乎呈线性增长，节点 3 以后增速下降。说明节点增加确实能提高效率，但是理论上增加一个节点提升 1 倍效率，在实际上很难达到。而对比两个数据集可以发现，数据集越大，加速效果越明显。

3.3　与其他推荐算法的对比实验

下面主要把本文算法(iMCF)与基于隐式反馈的矩阵分解算法(iMF)、基于用户的协同过滤算法(UBCF)、基于项目的协同过滤算法(IBCF)算法进行实验比较，得到各个阶段的数据。首先选取不同的 K 值（推荐队列的长度）进行实验，得出对应的准确率，具体结果如图 5 所示。然后比较这三种算法在相同情况下的 Rank 值，具体结果如图 6 所示。

通过图 5、6 的比较可以看出，当 K 值逐渐增大时，这三种算法的准确率逐渐下降，召回率在逐步增加。不同 K 值时 iMCF 算法都能得到最大的准确率和召回率，并且 iMCF 算法对于准确率和召回率的提升比较明显。这说明 iMCF 算法比其他两个算法的推荐质量更佳，准确度更好。

4 结束语

针对越来越庞大的网络资源和以往推荐算法的不足，本文提出了一种引入隐式反馈的多维度推荐算法(iMCF)。该算法充分利用了 Hadoop 集群的优势，有效结合了云模型，引入了隐式反馈数据，并且通过动态确定权重，使得目标用户对目标项目的预测评分更加精确。实验数据表明，该算法能适应大数据环境，并由于利用了云模型和三个维度数据，数据稀疏性问题也得到了合理的解决，推荐质量也上升了一个档次。而随着时间的变化，人们的兴趣也会发生改变，如何来衡量变化对推荐质量的影响，是笔者下个阶段的研究重点。

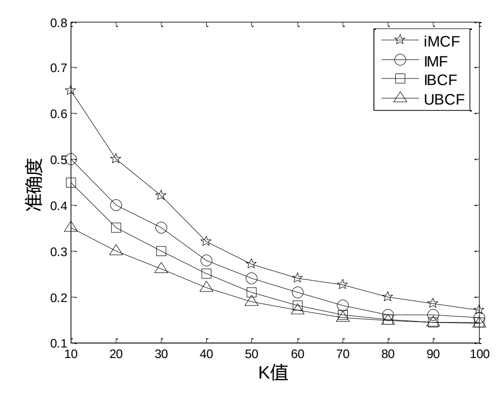


图 5 不同 K 值下其他算法与 iMCF 算法的 Precision 比较

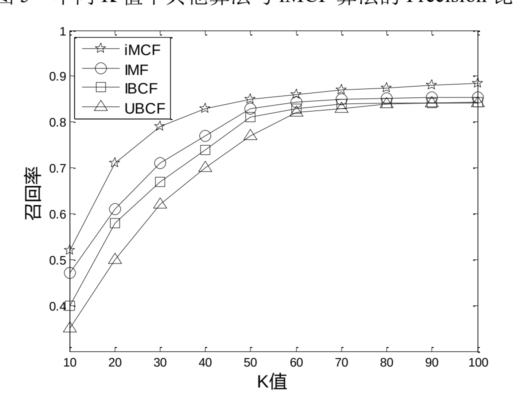


图 6 不同 K 值下其他算法与 iMCF 算法的 Rank 比较

参考文献：

[1] Liu Xin, Aberer K. SoCo: a social network aided context-aware recommender system [C]// Proc of International Conference on World Wide Web. [S. l.] : ACM Press, 2013: 781-802.

[2] Sun Guangfu, Wu Le, Liu Qi, *et al.* Recommendations based on collaborative filtering by exploiting sequential behaviors [J]. Journal of Software, 2014, 24 (11): 2721-2733.

[3] Zhang Jia, Lin Yaojin, Lin Menglei, *et al.* An effective collaborative filtering algorithm based on user preference clustering [J]. Applied Intelligence, 2016, 45 (2): 230-240.

[4] 于洪，李俊华. 一种解决新项目冷启动问题的推荐算法 [J]. 软件学报，2015, 26 (6): 1395-1408. (Yu Hong, Li Junhua. A recommended algorithm for solving cold start problems of new projects [J]. Journal of Software, 2015, 26 (6): 1395-1408.)

[5] 荣辉桂，火生旭，胡春华，等. 基于用户相似度的协同过滤推荐算法 [J]. 通信学报，2014, 35 (2): 16-24. (Rong Huigui, Huo Shengxv, Hu Chunhua, *et al.* Collaborative filtering recommendation algorithm based on user similarity [J]. Journal of Communications, 2014, 35 (2): 16-24.)

[6] Li Chenyang, He Kejing. CBMR: an optimized MapReduce for item-based collaborative filtering recommendation algorithm with empirical analysis [J]. Concurrency and Computation: Practice and Experience, 2017, 29 (10): 129-138.

[7] Ba Qilong, Li Xiaoyong, Bai Zhongying. Clustering collaborative filtering recommendation system based on SVD algorithm [C]// Proc of IEEE International Conference on Software Engineering and Service Science. [S. l.] : IEEE Press, 2013: 963-967.

[8] Hernando A, Bobadilla J, Ortega F. A non negative matrix factorization for collaborative filtering recommender systems based on a Bayesian probabilistic model [J]. Knowledge-Based Systems, 2016, 97 (9): 188-202.

[9] Yazdanfar N, Thomo A. Link recommender: collaborative-filtering for recommending URLs to twitter users [J]. Procedia Computer Science, 2013, 19 (4): 412-419.

[10] Park Y, Park S, Jung W, *et al.* Reversed CF: a fast collaborative filtering algorithm using a K-nearest neighbor graph [J]. Expert Systems with Applications, 2015, 42 (8): 4022-4028.

[11] 印鉴，王智圣，李琪，等. 基于大规模隐式反馈的个性化推荐 [J]. 软件学报，2014, 25 (9): 1953-1966. (Yin Jian, Wang Zhisheng, Li Qi, *et al.* Personalized recommendation based on large-scale implicit feedback [J]. Journal of Software, 2014, 25 (9): 1953-1966.)

[12] 王智圣，李琪，汪静，等. 基于隐式用户反馈数据流的实时个性化推荐 [J]. 计算机学报，2016, 39 (1): 52-64. (Wang Zhisheng, Li Qi, Wang Jing, *et al.* Real-time personalized recommendation based on implicit user feedback data stream [J]. Journal of Computer, 2016, 39 (1): 52-64.)

[13] Yuan Ting, Cheng Jian, Zhang Xi, *et al.* A weighted one class collaborative filtering with content topic features [C]// Proc of International Conference on Multimedia Modeling. Berlin: Springer, 2013: 417-427.

[14] Li Gai, Zhang Zhiqiang, Wang Liyang, *et al.* One-class collaborative filtering based on rating prediction and ranking prediction [J]. Knowledge-Based Systems, 2017, 124 (8): 46-54.

[15] 黄创光，印鉴，汪静，等. 不确定近邻的协调过滤推荐算法 [J]. 计算机学报，2010, 33 (8): 1369-1377. (Huang Chuang guang, Yin Jian, Wang Jing, *et al.* Coordinated filtering recommendation algorithm for uncertain neighbors [J]. Journal of Computer, 2010, 33 (8): 1369-1377.)

[16] Kumar N P, Fan Zhenzhen. Hybrid user-item based collaborative filtering [J]. Procedia Computer Science, 2015, 60 (1): 1453-1461.

[17] 陆艺，曹健. 面向隐式反馈的推荐系统研究现状与趋势 [J]. 计算机科学，2016, 43 (4): 7-15. (Lu Yi, Caojian. Research status and trends of recommender system for implicit feedback [J]. Computer Science, 2016, 43 (4): 7-15.)

[18] 燕彩蓉，张青龙，赵雪，等. 基于广义高斯分布的贝叶斯概率矩阵分解

方法 [J]. 计算机研究与发展, 2016, 52 (12): 2793-2800. (Yan Cairong, Zhang Qinlong, Zhao Xue, *et al.* Bayesian probability matrix decomposition method based on generalized Gaussian distribution [J]. Journal of Computer Research and Development, 2016, 52 (12): 2793-2800)

[19] Bauer J, Nanopoulos A. Recommender systems based on quantitative implicit customer feedback [J]. Decision Support Systems, 2014, 68 (6): 77-88.